Using Case Based Heuristics to Speed up Reinforcement Learning

Reinaldo A. C. Bianchi
Centro Universitário da FEI
São Bernardo do Campo, Brazil.
rbianchi@fei.edu.br

Raquel Ros and **Ramon Lopez de Mantaras**
Artificial Intelligence Research Institute (IIIA-CSIC)
Bellaterra, Spain.
{ros, mantaras}@iiia.csic.es

Abstract

The aim of this work is to combine three successful AI techniques –Reinforcement Learning (RL), Heuristics Search and Case Based Reasoning (CBR)– creating a new algorithm that allows the use of cases in a case base as heuristics to speed up Reinforcement Learning algorithms. This approach, called Case Based Heuristically Accelerated Reinforcement Learning (CB-HARL), builds upon an emerging technique, the Heuristic Accelerated Reinforcement Learning (HARL), in which RL methods are accelerated by making use of heuristic information. Empirical evaluations were conducted in a simulator for the RoboCup Four-Legged Soccer Competition, and the results obtained show that using CB-HARL, the agents learn faster than using either RL or HARL methods.

Introduction

Reinforcement Learning (RL) (Sutton and Barto 1998) is a very successful Artificial Intelligence sub-area. It is concerned with the problem of learning from interaction to achieve a goal. Given an autonomous agent interacting with its environment via perception and action, on each interaction step the agent senses the current state s of the environment, and chooses an action a to perform. The action a alters the state s of the environment, and a scalar reinforcement signal r (a reward or penalty) is provided to the agent to indicate the desirability of the resulting state. The policy π is some function that tells the agent which actions should be chosen, and it is learned through trial-and-error interactions of the agent with its environment.

RL algorithms are very useful for solving a wide variety problems when the model is not known in advance, with many algorithms possessing guarantees of convergence to equilibrium (Watkins 1989; Sutton and Barto 1998). Unfortunately, the convergence of any RL algorithm may only be achieved after an extensive exploration of the state-action space, which is usually very time consuming.

One way to speed up the convergence of RL algorithms is by making use of a heuristic function in a manner similar to the use of heuristics in informed search algorithms. Heuristically Accelerated Reinforcement Learning (HARL) methods, which have been recently proposed (Bianchi, Ribeiro, and Costa 2008), apply a conveniently chosen heuristic function for selecting the appropriate actions to perform in order to guide exploration during the learning process. Although several methods have been successfully applied for defining the heuristic function, a very interesting option has not been explored yet: the reuse of previously learned policies, using a Case Based Reasoning approach to define an heuristic function.

Case Based Reasoning (Aamodt and Plaza 1994; Lopez de Mantaras et al. 2005) is an AI technique that has been shown to be useful in a multitude of domains, with widespread applications ranging from the diagnosis and treatment of many medical problem to the synthesis of high quality expressive music. CBR uses knowledge of previous situations (cases) to solve new problems, by finding a similar past case and reusing it in the new problem situation. In the CBR approach, a case usually describes a problem and its solution, i.e., the state of the world in a given instant and the sequence of actions to perform to solve that problem.

This paper investigates the combination of Case Based Reasoning (CBR) and Heuristically Accelerated Reinforcement Learning (HARL) techniques, with the goal of speeding up RL algorithms by using previous domain knowledge as heuristics, stored as a case base. To do so, we propose a new algorithm, the Case Based Heuristically Accelerated Q–Learning (CB-HAQL), which incorporates Case Based Reasoning techniques into an existing HARL algorithm, the Heuristically Accelerated Q–Learning (HAQL). The results shown in this paper were first published at the ICCBR'09 Conference (Bianchi, Ros, and Lopez de Mantaras 2009).

Heuristically Accelerated Reinforcement Learning

A Heuristically Accelerated Reinforcement Learning (HARL) algorithm (Bianchi, Ribeiro, and Costa 2008) is a way to solve a MDP problem with explicit use of a heuristic function $\mathcal{H} : \mathcal{S} \times \mathcal{A} \rightarrow \Re$ for influencing the choice of actions by the learning agent. $H(s, a)$ defines the heuristic that indicates the importance of performing action a when visiting state s. The heuristic function is strongly associated with the policy indicating which action must be taken regardless of the action-value of the other actions that could

Copyright © 2010, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

Table 1: The HAQL algorithm.

Initialize $\hat{Q}_t(s,a)$ and $H_t(s,a)$ arbitrarily.
Repeat (for each episode):
  Initialize s.
  Repeat (for each step):
    Update the values of $H_t(s,a)$ as desired.
    Select an action a using equation 1.
    Execute the action a, observe $r(s,a)$, s'.
    Update the values of $Q(s,a)$ according to equation 3.
    $s \leftarrow s'$.
  Until s is terminal.
Until some stopping criterion is reached.

be used in the state.

The first HARL algorithm proposed was the Heuristically Accelerated Q–Learning (HAQL) (Bianchi, Ribeiro, and Costa 2008), as an extension of the Q–Learning algorithm (Watkins 1989). The only difference between the two algorithms is that in the HAQL makes use of an heuristic function $H(s,a)$ in the $\epsilon - greedy$ action choice rule, that can be written as:

$$\pi(s) = \begin{cases} \arg\max_a \left[\hat{Q}(s,a) + \xi H(s,a)^\beta\right] & \text{if } q \leq p, \\ a_{random} & \text{otherwise,} \end{cases} \quad (1)$$

where $H(s,a)$ is the heuristic function that plays a role in the action choice, ξ and β are design parameters that control the influence of the heuristic function, q and p are parameters that define the exploration/exploitation tradeoff and a_{random} is an action randomly chosen among those available in state s.

As a general rule, the value of $H(s,a)$ used in HAQL should be higher than the variation among the $\hat{Q}(s,a)$ values for the same $s \in \mathcal{S}$, in such a way that it can influence the choice of actions, and it should be as low as possible in order to minimize the error. It can be defined as:

$$H(s,a) = \begin{cases} \max_i \hat{Q}(s,i) - \hat{Q}(s,a) + \eta \text{ if } a = \pi^H(s), \\ 0 \text{ otherwise.} \end{cases} \quad (2)$$

where η is a small real value (usually 1) and $\pi^H(s)$ is the action suggested by the heuristic policy.

The Q-values are updated using the Q-learning equation:

$$\hat{Q}(s,a) \leftarrow \hat{Q}(s,a) + \alpha \left[r + \gamma \max_{a'} \hat{Q}(s',a') - \hat{Q}(s,a)\right], \quad (3)$$

where s is the current state; a is the action performed in s; r is the reward received; s' is the new state; γ is the discount factor ($0 \leq \gamma < 1$); and α is the learning rate.

Convergence of the HAQL algorithm was presented by Bianchi, Ribeiro and Costa (Bianchi, Ribeiro, and Costa 2008), together with the definition of an upper bound for the error in the estimation of Q. The complete HAQL algorithm is presented in Table 1.

Case Definition

The case definition used in this work is the one proposed by Ros (2009), which is composed of three parts: the problem description (P), the solution description (A) and the case scope (K), and it is formally described as a 3-tuple:

$$case = (P, A, K).$$

The problem description P corresponds to the situation in which the case can be used. For example, for a robotic soccer problem, the description of a case can include the robot position, the ball's position and the positions of the other robots in the game. For a game with n robots (teammates and opponents), P can be:

$$P = \{x_B, y_B, x_{R_1}, y_{R_1}, \ldots, x_{R_n}, y_{R_n}\}.$$

The solution description is composed by the sequence of actions that each robot must perform to solve the problem, and can be defined as:

$$A = \{R_1 : [a_{1_1}, a_{1_2}, ..., a_{1_{p_1}}], \ldots, R_m : [a_{m_1}, a_{m_2}, ..., a_{m_{p_m}}]\},$$

where m is the number of robots in the team, a_{i_j} is an individual or joint action that robot R_i must perform and p_i corresponds the number of actions the robot R_i performs.

The case scope defines the applicability boundaries of the cases to be used in the retrieval step. In the case of a robot soccer problem, K can be represented as ellipsoids centered on the ball's and the opponents' positions indicated in the problem description. It can be defined as:

$$K = \{(\tau^x_B, \tau^y_B), (\tau^x_{R_1}, \tau^y_{R_1}) \ldots, (\tau^x_{R_l}, \tau^y_{R_l})\},$$

where τ^x_B, τ^y_B corresponds to the x and y radius of the ellipsoid region around the ball and $(\tau^x_{R_1}, \tau^y_{R_1}) \ldots, (\tau^x_{R_l}, \tau^y_{R_l})$ the radius of the regions around the l opponent robots in the game.

Case retrieval is in general driven by a similarity measure between the new problem and the solved problems in the case base. In this work we use the case retrieval method proposed by Ros (2009), which considers the similarity between the problem and the case, the cost of adapting the problem to the case, and the applicability of the solution of the case. These functions and the complete case retrieval algorithm are described in detail in Ros (2009).

Combining Case Based Reasoning and Reinforcement Learning

In order to provide HARL algorithms the capability of reusing previous knowledge from a domain, we propose a new algorithm, the Case Based HAQL, which extends the HAQL algorithm with the abilities to retrieve a case stored in a base, adapt it to the current situation, and build a heuristic function that corresponds to the case.

Inside this HAQL main loop, before the action selection is made, we added steps to compute the similarity of the cases with the current state and the cost of adaptation of these cases. A case is retrieved if the similarity is above a certain threshold, and the adaptation cost is low. After a case is retrieved, an heuristic is computed using Equation 2 and the sequence of actions suggested by the case selected. This heuristic is used for a certain amount of time, equal to the number of actions of the retrieved case. After that time,

Table 2: The CB-HAQL algorithm.

Initialize $\hat{Q}_t(s,a)$ and $H_t(s,a)$ arbitrarily.
Repeat (for each episode):
  Initialize s.
  Repeat (for each step):
    Compute similarity and cost.
    If there is a case that can be reused:
      Retrieve and Adapt if necessary.
      Compute $H_t(s,a)$ using Equation 2 with the actions suggested by the case selected.
    Select an action a using equation 1.
    Execute the action a, observe $r(s,a)$, s'.
    Update the values of $Q(s,a)$ according to equation 3.
    $s \leftarrow s'$.
  Until s is terminal.
Until some stopping criterion is reached.

a new case can be retrieved. The complete CB-HAQL algorithm is presented in Table 2.

Although this is the first work that combines CBR with RL using an explicit heuristic function, this is not the first work on combining the both fields. Drummond (2002) was probably the first to use CBR to speed up RL, proposing to accelerate RL by transferring parts of previously learned solutions to a new problem. Sharma *et al.* (2007) make use of CBR as an approximation function for RL, and RL as a revision algorithm for CBR in a hybrid architecture system; Juell and Paulson (2003) exploit the use of RL to learn similarity metrics and Auslander *et al.* (2008) use CBR to adapt quickly an RL agent to changing conditions of the environment.

Our approach differs from all previous research combining CBR and RL because it makes use of cases only as heuristics. If the case base contains a case that can be used in a given situation, then there will be a speed up in the convergence time. But if the case base does not contain any useful case –or even if it contains cases that implement wrong solutions to the problem– the agent will still learn the optimal solution by using the RL component of the algorithm.

Experiments in the Robotic Soccer Domain

Empirical evaluations of the CB-HAQL approach were carried out in an extended version of the PuppySim 2 simulator (Ros et al. 2009). This simulator represents the basic aspects of the RoboCup Standard Platform League, Four-Legged Soccer Competition.

Using this simulator experiments were performed using two attackers against a defender and a goalie. The attackers are two robots controlled by one of the algorithms to be evaluated: the Q–Learning, the HAQL or the CB-HAQL and we have also compared them to the results of the CBR system used by Ros (2009). The opponents perform the same reactive behavior when playing against any of the evaluated approaches. The defender and the goalie have a home region which cannot go beyond. If the ball is within its home region, then the robot moves towards the ball and clears it. Otherwise, the robot remains in the boundary of its home region, facing the ball to maintain it in view.

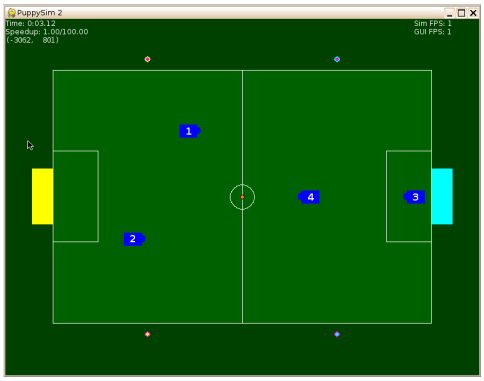

Figure 1: The PuppySim2 users' interface showing the robots at one possible initial positions.

Each trial begins with the attackers being positioned in the field in a random position, and the defender, the goalie and the ball in a fixed location (ball in the center, and defender and goalie in the center of their home region). Figure 1 shows the PuppySim 2 users' interface with one starting configuration. A trial ends when either the attackers score a goal, the ball goes out of the field or the goalie touches it. The heuristic used in the HAQL algorithm was defined using a simple rule: if holding the ball, go to the opponents goal, not taking into account the opponents positions, leaving the task of how to divert the opponent to the learning process. The heuristic used in the CB-HAQL is computed during the games. The case base used for the experimentation is composed of 136 cases, which cover the most significant situations that can occur in the evaluation presented in this work. From this set, 34 cases are initially defined, while the remaining ones are automatically generated using spatial transformations exploiting the symmetries of the soccer field. The reward the agents receive are the same for all algorithms: -100 every time the ball goes out of the field or the goalie touches it, and $+100$ a robot scores a goal.

In order to evaluate each trial we classify the possible outcomes as: "goal" (the ball enters the goal), "close" (the ball goes out of the field but passes at less than 25cm of the goal-post), "block" (the goalie stops or kicks the ball) and "out" (the ball goes out the field without being a goal or close to goal). We also consider the "to-goal" balls, which correspond to balls that are either goals or close to goal. This measure indicates the degree of goal intention of the kicks. Thus, although the balls might not enter the goal, at least they were intended to do so.

Twenty five training sessions were run for the three algorithms, with each session consisting of 1000 trials. The parameters used in the experiments were the same for all the algorithms: $\alpha = 0,9$, the exploration/ exploitation $= 0.2$, $\gamma = 0.9$ and $\eta = 1$. HAQL parameters ξ and β are set to 1. Values in the Q table were randomly initiated.

Table 3 summarizes the ball classification outcome obtained (results in percentage) using the CBR approach and the three learning algorithms. The results for the CBR approach are the average of 500 trials, and the results for the Q–learning, HAQL and CB-HAQL are the average of 100

Table 3: Ball outcome classification (results in percentage).

Approach	Goal	Close	To-Goal	Blocked	Out
CBR	35	5	40	38	22
Q–Learning	2	2	4	22	74
HAQL	16	4	20	20	60
CB-HAQL	40	7	47	36	17

trials, using the Q-table that the three algorithms had at the end of the 1000^{th} trial. As we can see the percentage of balls to goal with the CB-HAQL approach is higher compared to either the HAQL or the Q-Learning algorithms. Moreover, the percentage of balls out are lower when using CBR and CB-HAQL, indicating that the defender had less opportunities to take the ball and kick it out of the field, and that the agent performed less random exploration.

Finally, Figure 2 shows the learning curves for all algorithms presenting the percent of goals scored by the learning team. It is possible to verify that at the beginning of the learning phase HAQL has worse performance than the CB-HAQL, and as the trials proceed, the performance of both algorithms become similar, as expected, since all the algorithms converge to equilibrium. The Q–learning is clearly the one with the worst performance, since it takes much more trials for it to start to learn even basic policies, as not to kick the ball out of the field. In this figure it can also be observed the constant performance of two agents using only the case-based approach (i.e. without learning). Student's t–test was used to verify the hypothesis that the use of heuristics speeds up the learning process. Using the data from Fig. 2, the result is that the CB-HAQL is better (makes more goals) than HAQL and Q–Learning until the 300^{th} trial, with a level of confidence greater than 95%.

Conclusion

This work presented a new algorithm, called Case Based Heuristically Accelerated Q–Learning (CB-HAQL), which allows the use of a case base to define heuristics to speed up the well-known Reinforcement Learning algorithm Q–Learning. This approach builds upon an emerging technique, the Heuristic Accelerated Reinforcement Learning (HARL), in which RL methods are accelerated by making use of heuristic information.

The experimental results obtained showed that CB-HAQL attained better results than HAQL and Q–Learning for the domain of robotic soccer games. For example, the Q–Learning, after 1000 learning trials, still could not produce policies that scored goals on the opponent, while the HAQL was able to score some goals but significantly less than the CBR alone and the CB-HAQL. Another interesting finding is that the number of goals scored by the CB-HAQL after 1000 trials was even slightly higher than the number of goals scored by the CBR approach alone, indicating that the learning component of the CB-HAQL algorithm was able to improve the initial case base.

Finally, since Heuristic functions allow RL algorithms to solve problems where the convergence time is critical, as in many real time applications, in future work we plan to incorporate CBR in other well known RL algorithms, like SARSA, Q(λ), Minimax-Q, and Nash-Q.

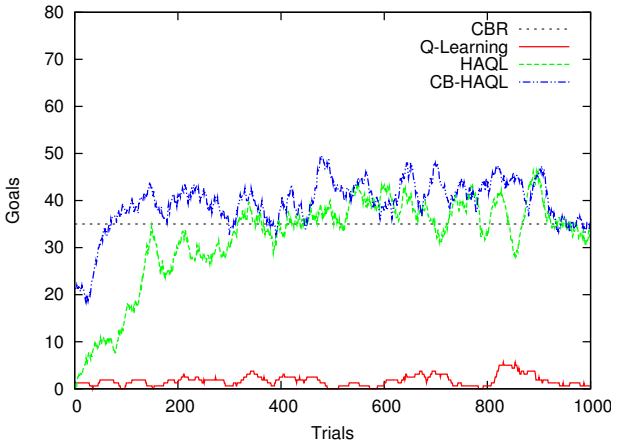

Figure 2: Percentage of goals scored in each trial using the CBR (constant line at 35%), Q–learning, the HAQL and the CB-HAQL algorithms.

References

Aamodt, A., and Plaza, E. 1994. Case-based reasoning: foundational issues, methodological variations, and system approaches. *AI Commun.* 7(1):39–59.

Auslander, B.; Lee-Urban, S.; Hogg, C.; and Muñoz-Avila, H. 2008. Recognizing the enemy: Combining reinforcement learning with strategy selection using case-based reasoning. In *ECCBR*, 59–73. Berlin:Springer-Verlag.

Bianchi, R. A. C.; Ribeiro, C. H. C.; and Costa, A. H. R. 2008. Accelerating autonomous learning by using heuristic selection of actions. *Journal of Heuristics* 14(2):135–168.

Bianchi, R. A. C.; Ros, R.; and Lopez de Mantaras, R. 2009. Improving reinforcement learning by using case based heuristics. In *International Conference on Case-Based Reasoning*, 75–89.

Lopez de Mantaras, R.; McSherry, D.; Bridge, D.; Leake, D.; Smyth, B.; Craw, S.; Faltings, B.; Maher, M. L.; Cox, M. T.; Forbus, K.; Keane, M.; Aamodt, A.; and Watson, I. 2005. Retrieval, reuse, revision and retention in case-based reasoning. *Knowl. Eng. Rev.* 20(3):215–240.

Drummond, C. 2002. Accelerating reinforcement learning by composing solutions of automatically identified subtasks. *JAIR* 16:59–104.

Juell, P., and Paulson, P. 2003. Using reinforcement learning for similarity assessment in case-based systems. *IEEE Intel. Sys* 18(4):60–67.

Ros, R.; Arcos, J. L.; Lopez de Mantaras, R.; and Veloso, M. 2009. A case-based approach for coordinated action selection in robot soccer. *AIJ* 173(9-10):1014–1039.

Sharma, M.; Holmes, M.; Santamaría, J. C.; Irani, A.; Jr., C. L. I.; and Ram, A. 2007. Transfer learning in real-time strategy games using hybrid cbr/rl. In *IJCAI*, 1041–1046.

Sutton, R. S., and Barto, A. G. 1998. *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press.

Watkins, C. J. C. H. 1989. *Learning from Delayed Rewards*. Ph.D. Dissertation, University of Cambridge.